Visualizing Android Feature Maliciousness through time

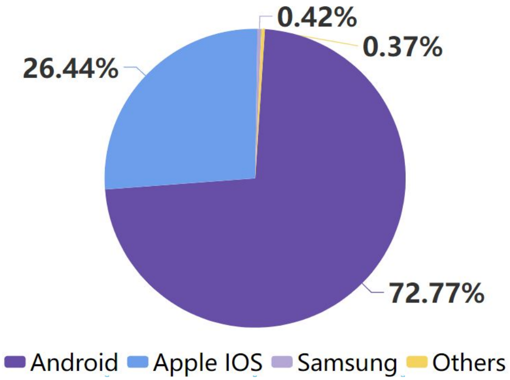
Michael Tegegn

Content

1. Introduction and Motivation
    a. Android
    b. Android Malware Detection
2. Visualizing Android Features over time
    a. Related Work
    b. Dataset and Abstraction
    c. Design Decisions
    d. Demo
3. Limitations and Future work

Android

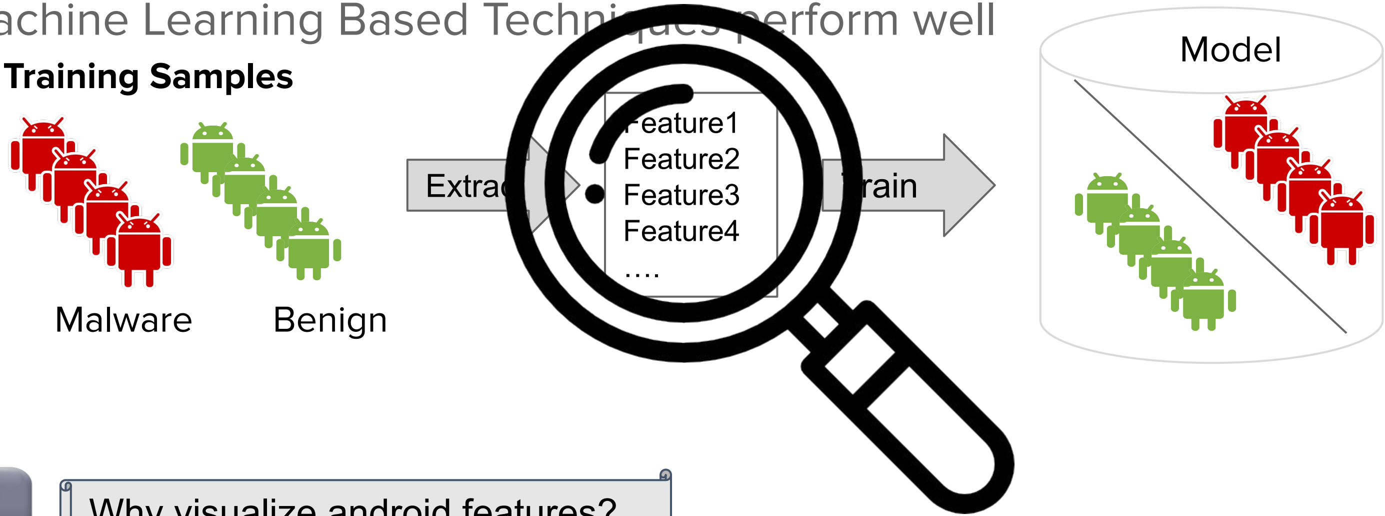
- Most widely used Mobile OS



- 12,000 new Android malware instances every day. unb

Android Malware Detection

- Machine Learning Based Techniques perform well



Training Samples

Malware

Benign

Extrac

Feature1
Feature2
Feature3
Feature4
....

rain

Model

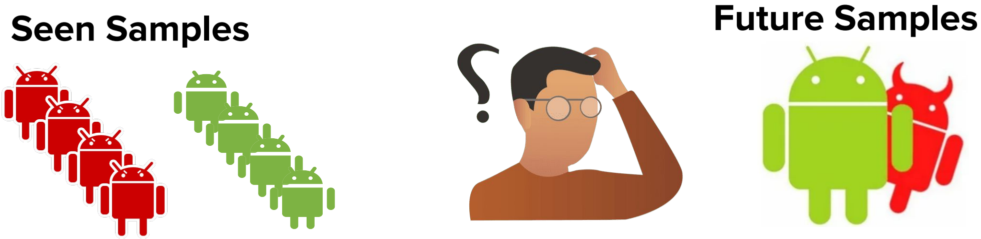
Why visualize android features?

- Aid in Feature Selection
  - Eg. Selecting features more common in malware applications can help boost robustness [1]

[1] “On Benign Features in Malware Detection” by Michael Cao et al, ASE, 2020

Sample Selection: Which samples to train on?

- We want to identify future malware



- We can just train on all existing benign and malware samples

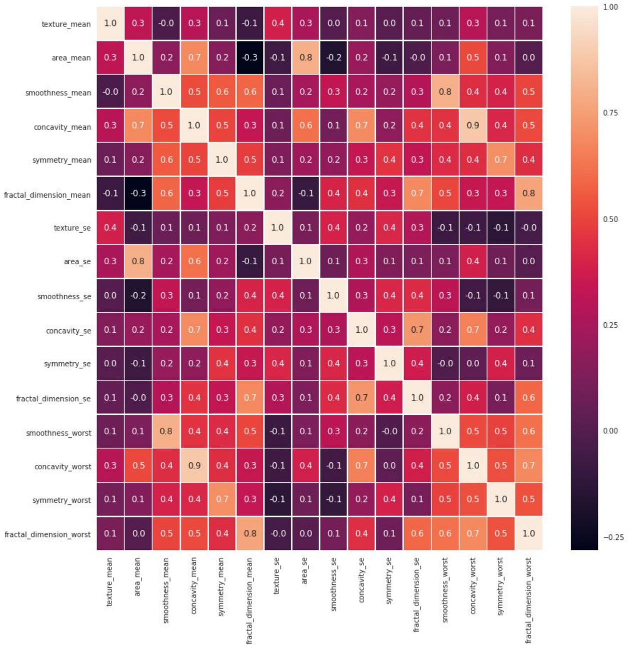
Why visualize android features through time?

- Identify features that can help detect future malware [2]

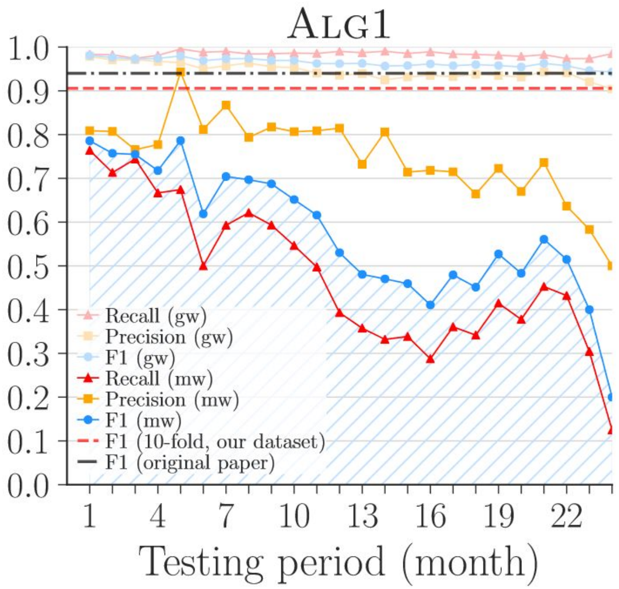
[2] “TESSERACT: Eliminating Experimental Bias in Malware Classification across Space and Time. by Pendlebury et al, USENIX, 2019

Related Work: Visualizations

1. Correlation Between Features



2. Model Performance through time



Data:

- Android Applications represented as a set of (DREBIN) features
    - Benign and Malware Android applications
        - Eg. app1: { feature1, feature3, feature5 } label: benign
- Feature selection metrics and their results
    - Features ranked according the feature selection metrics
        - Eg. Mutual info metric: {feature3, feature1, feature2}

Derived Attributes:

$$Maliciousness\ of\ \ Feature\ x = \frac{Malicious\ Apps\ with\ x\ -\ Benign\ Apps\ with\ x}{Total\ Number\ of\ \ Apps\ with\ x}$$

- -1 => x only in malware apps
- 1 => x only in benign apps

$$Normalized\ freq\ of\ \ Feature\ x = \frac{Apps\ with\ Feature\ x}{Total\ Number\ of\ \ Apps}$$

- 1 => All apps contain x
- 0 => No apps contain x

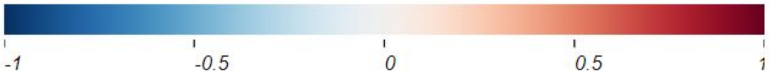
Data Abstraction Summary

Attribute	**Kind**
Feature	Categorical
Feature Set / Feature Family	Categorical
App Development Year	Ordinal
Feature Maliciousness	Quantitative
Feature Normalized Frequency	Quantitative

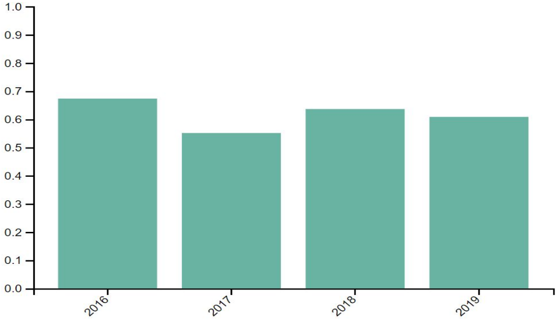
Goal: Identify feature trends in android applications
Target Group: Malware detection tool developers

Visualization Design: Encodings

- Feature Maliciousness over time using heatmap
  - **y-axis:** Feature **x-axis**: Development year
  - Maliciousness encoded by **Blue-Red Diverging Scale**



- Normalized frequency over time using bar charts
  - **y-axis:** Feature **x-axis**: Development year
  - **Bar height**: Normalized frequency of a feature

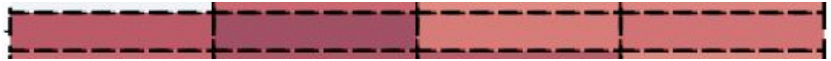


Normalized Frequency per year of selected feature

usedpermissionslist_android.permission.internet

Visualization Design: View Manipulation

- **Requirement 1**: Accommodate for large number of features
  - Design Solution
    - **Filter** features based on feature set
    - **Order** features based on popularity, maliciousness, ...
    - **Alter view** using scrolling
- **Requirement 2**: Show selected features
  - Design solution
    - Show selection using **dashed border lines**



- **Requirement 3**: Display normalized frequency of a feature on demand
  - Design solution
    - Add side view bar chart upon selection

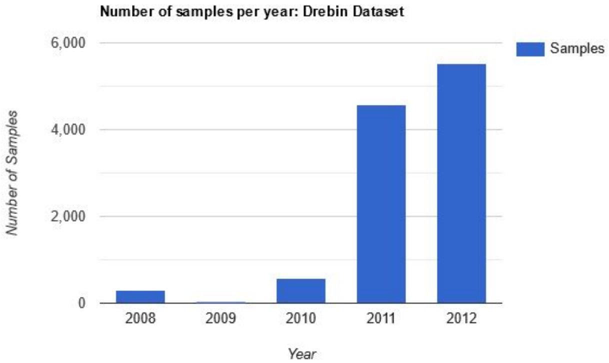
Demo

http://127.0.0.1:5500/index.html

Limitations and Future Work

Limitations:

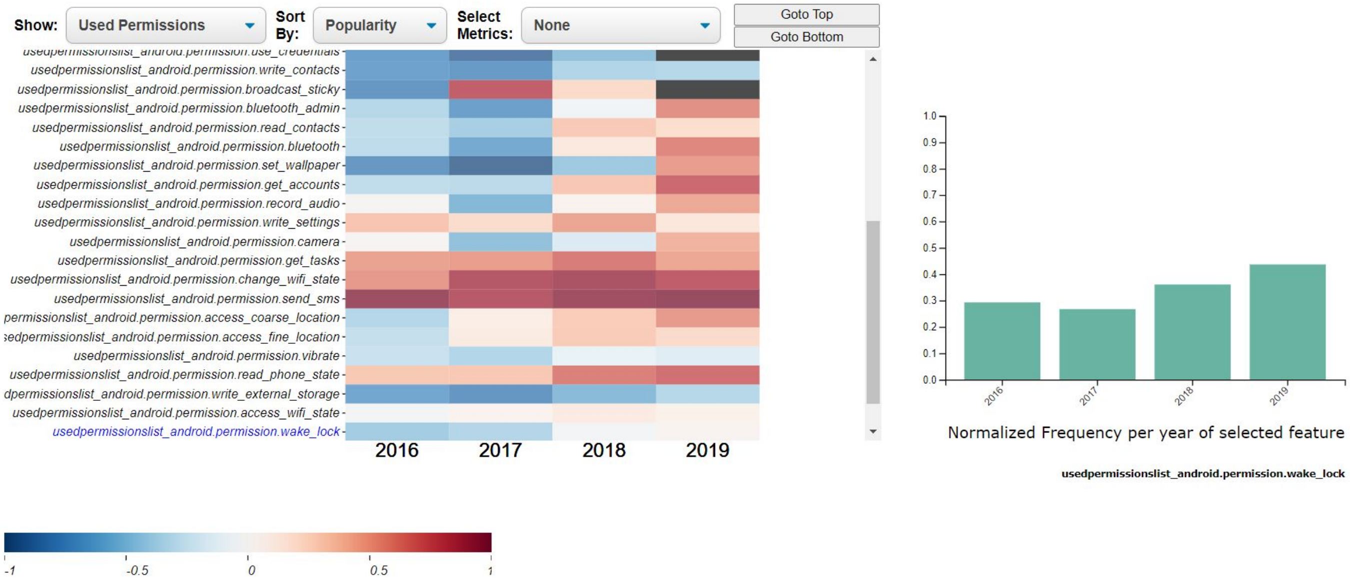
- Sampling bias in dataset
  - Eg. Drebin Dataset:



- Short time duration for the two datasets (VT: 4 years, DREBIN: 5 years)
- Still a large number of features (Scrolling required)

Future work:

- Hand select features and train models directly on the vis interface
- Extend for any domain that requires analysis of features for feature selection



Show: Used Permissions

Sort By: Popularity

Select Metrics: None

Goto Top
Goto Bottom
usedpermissionslist_android.permission.use_credentials
usedpermissionslist_android.permission.write_contacts
usedpermissionslist_android.permission.broadcast_sticky
usedpermissionslist_android.permission.bluetooth_admin
usedpermissionslist_android.permission.read_contacts
usedpermissionslist_android.permission.bluetooth
usedpermissionslist_android.permission.set_wallpaper
usedpermissionslist_android.permission.get_accounts
usedpermissionslist_android.permission.record_audio
usedpermissionslist_android.permission.write_settings
usedpermissionslist_android.permission.camera
usedpermissionslist_android.permission.get_tasks
usedpermissionslist_android.permission.change_wifi_state
usedpermissionslist_android.permission.send_sms
permissionslist_android.permission.access_coarse_location
sedpermissionslist_android.permission.access_fine_location
usedpermissionslist_android.permission.vibrate
usedpermissionslist_android.permission.read_phone_state
dpermissionslist_android.permission.write_external_storage
usedpermissionslist_android.permission.access_wifi_state
usedpermissionslist_android.permission.wake_lock
2016
2017
2018
2019
-1
-0.5
0
0.5
1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
2016
2017
2018
2019
Normalized Frequency per year of selected feature
usedpermissionslist_android.permission.wake_lock

Thank you!

Questions?